Vitor Amadeu Souza

Obtendo a leitura de

Sensores

no smartphone via

USB OTG

Programado no Arduino

© 2020 by Cerne Tecnologia e Treinamento Ltda.

© 2020 by Vitor Amadeu Souza

Nenhuma parte desta publicação poderá ser reproduzida sem autorização prévia e escrita de **Cerne Tecnologia e Treinamento Ltda.** Este livro publica nomes comerciais e marcas registradas de produtos pertencentes a diversas companhias. O editor utiliza as marcas somente para fins editoriais e em benefício dos proprietários das marcas, sem nenhuma intenção de atingir seus direitos.

Maio de 2020

Direitos reservados por:
Cerne Tecnologia e Treinamento Ltda

Produção: *Cerne Tecnologia e Treinamento*
E-mail da Empresa: *cerne@cerne-tec.com.br*
Home Page: *www.cerne-tec.com.br.com.br*
Atendimento ao Consumidor: *sac@cerne-tec.com.br*
Contato com o Autor: *vitor@cerne-tec.com.br*



FEITO NO BRASIL

“Quem se vinga depois da vitória é indigno de vencer.”

Voltaire

Cerne Tecnologia

A Cerne Tecnologia tem uma equipe preparada para desenvolvimento de projetos eletrônicos em diversas áreas: Médica, Entretenimento, Industrial, Robótica, Científica, Automobilística, Aeronáutica, etc. Trabalhamos com tecnologia microcontrolada usando o PIC, ARM, AVR, 8051, dsPIC, PIC24, PIC32 além do Arduino, Raspberry, Beaglebone etc. Desenvolvemos o projeto desde sua concepção até a entrega do produto final, passando pelas etapas de esquema elétrico, protótipo e desenvolvimento de circuito impresso.

Desenvolvemos aplicativos para smartphones/tablets Android, iOS, Blackberry, Windows Phone e no desenvolvimento de softwares a nível PC para plataforma Windows, usando ferramentas como o Visual Basic, C# e C++.

Atuamos na parte de montagem de placas, onde podemos fornecer ambos os serviços de desenvolvimento de projetos e produção ou apenas um destes.

Desenvolvemos esquemas elétricos e layout de PCI, tanto em tecnologia convencional como SMD.

Temos a flexibilidade de customizar um de nossos produtos, de modo a atender a uma necessidade específica do cliente, tornando o custo de desenvolvimento menor se comparado a construção de um projeto desde a sua fase inicial.

Desenvolvemos e fornecemos kits didáticos para diversos microcontroladores além de apostilas, livros e e-books.

Na hora de desenvolver um projeto ou equipar seu laboratório não hesite em nos contatar. Entre em contato conosco através do endereço cerne-tec.com.br para obter mais informações.

Cerne
Conhecimento para o Desenvolvimento

cerne-tec.com.br

Sumário

01. Metodologia de desenvolvimento **6**

02. Hardware e Software Arduino **7**

03. Ligando um Led no Arduino **10**

04. Transmissão Serial **17**

05. Display LCD **21**

06. Medindo Etanol em ppm **25**

07. Medindo a concentração de Amônia **38**

08. Amperímetro **55**

09. Barômetro **68**

10. Medindo a concentração de Benzeno **78**

11. Célula de carga **94**

12. Medição de Monóxido de Carbono **109**

13. Medição de Dióxido de Carbono **128**

14. Decibelímetro **147**

15. Dinamômetro **165**

16. Medindo o fluxo de massa **176**

17. Gaussímetro **190**

18. Medindo a concentração de GLP **206**

19. Medindo a concentração de H_2 **222**

20. Higrômetro **239**

21. Medindo a concentração de Ozônio **248**

22. pHmetro **266**

23. Voltímetro **282**

24. Termômetro **293**

25. Medindo a radiação UV **301**

26. Medição de volume de fluido por Ultrassom **319**

Capítulo 1

Metodologia de desenvolvimento

1. Introdução

Esta literatura é uma continuação da obra *Arduino – Prático e Objetivo* (2011) e *Programação para Arduino – Avançado* (2014) do mesmo autor e editora, onde o objetivo é desenvolver projetos que leiam algum dos sensores mencionados no sumário da obra para em seguida enviar para o smartphone Android através do app Serial USB Terminal via cabo USB OTG o valor mensurado, de modo que seja verificado o valor de acordo com o resultado da medição.

A placa didática utilizada foi a Arduino UNO, onde tal kit está à venda no site www.cerne-tec.com.br.

Capítulo 2
Hardware e Software Arduino

1. Introdução

O Arduino é uma ferramenta livre tanto em nível de hardware quanto de software, onde neste você encontra os recursos necessários para iniciar seus projetos, sem ser especialista nem em eletrônica quanto em software. Neste livro, são apresentados alguns exemplos de modo a chegar à proposta da obra que fornece uma base teórica de programação e hardware para você criar seus próprios projetos. Estes projetos podem ser úteis em disciplinas como robótica educacional quanto na criação de automatismos em geral, usados em projetos de escolas técnicas e faculdades.

2. Conhecendo o Software

O software usado neste livro foi o Arduino versão 1.6.6, que pode ser baixado gratuitamente no endereço abaixo:

http://arduino.cc/en/Main/Software

A grande vantagem deste software é o fato do mesmo funcionar em sistemas operacionais do tipo Windows, Linux e MacOS. A vantagem em usar o Linux é dispor de uma ferramenta gratuita que facilite o acesso aos alunos para realização de seus projetos.

Após realizar o download do software basta realizar a instalação do arquivo setup. Na figura abaixo, é apresentado à tela deste programa, que voltará a ser comentado em breve.



3. Conhecendo o Hardware

Existem diversas placas que podem ser usadas para realizar os experimentos. Nas figuras abaixo, são apresentadas algumas destas.



Arduino Uno



Arduino Ethernet

Existem outros modelos e todos estes, estão disponíveis para visualização e compra no endereço abaixo:

http://arduino.cc/en/Main/Hardware

Este livro utiliza a placa didática Arduino UNO como referência.

Capítulo 3

Ligando um Led no Arduino

1. Montando o Hardware

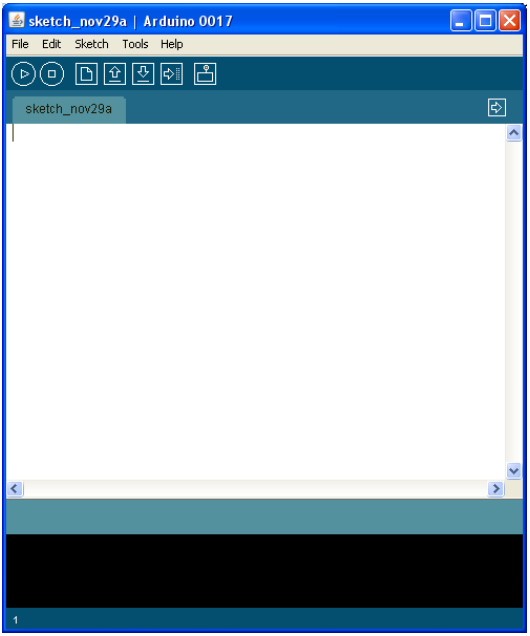
Neste primeiro exemplo o objetivo é ligar um led que está conectado à porta 13 do Arduino. Na figura abaixo há uma figura de um led.



Existem vários tipos de leds, com diferentes cores, sendo que isto não interfere no exemplo. Note que o led é composto de dois terminais, sendo que o maior é o terminal positivo enquanto que o menor o negativo. Isso deve ser observado com atenção, pois se invertido, o led não funcionará. Além disso, é necessário ligar um resistor em série com o led, de forma a não queimá-lo. Um resistor típico pode ter sua resistência de 220 Ω a 470 Ω. Ligue neste caso o terminal positivo do led ao pino 13 e o negativo ao GND da placa.

2. Programando o Arduino

Abra o programa Arduino, teremos a seguinte tela:



A estrutura básica para montarmos um programa, é a seguinte:

```
void setup()
{

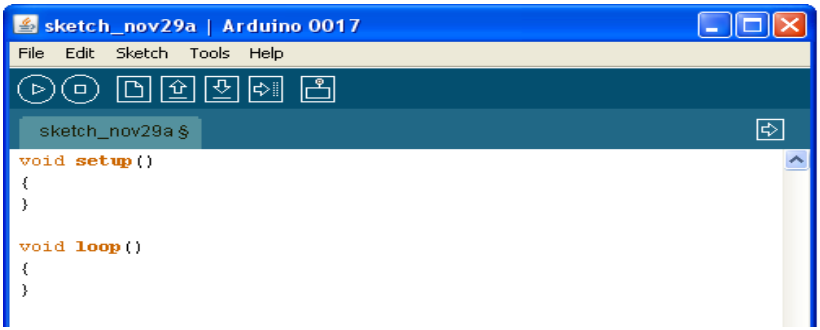
}

void loop()
{

}
```

Dentro das chaves do `void setup()` será feita a configuração básica do Arduino. Entenda este bloco como a BIOS do computador, em que realiza-se todas as configurações iniciais para que o mesmo venha a funcionar em seguida. Dentro das chaves do bloco void loop(), é feita a lógica de programação que deseja-se que a placa execute.

Digite o pequeno trecho de código no software Arduino, o resultado será como o apresentado a seguir:



Observe que este bloco refere-se a estrutura básica do programa, ainda não foi implementado propriamente dito, ligar o led que está conectado ao pino 13.

Um pino digital pode ser configurado tanto entrada quanto saída, ou seja, se ligar um led ao pino, ele será uma saída para o chip e se ligar um botão, será uma entrada. Como o led fica conectado a porta 13, deve-se configurar esta porta como saída e

para isso, utiliza-se a função `pinMode`, que permite configurar um pino como entrada ou saída. A seguir, é apresentada a configuração desta função.

```
pinMode(pino, modo);
```

Onde na opção pino informa-se qual pino deseja-se configurar e em modo, se ele será configurado como entrada ou saída, onde informa-se INPUT ou OUTPUT respectivamente. Sendo assim, o programa ficará da forma abaixo, em que o pino 13 está configurado como saída:

```
void setup()
{
     pinMode(13,OUTPUT);
}

void loop()
{

}
```

O pino está configurado como saída, mais ainda não foi ligado. Para fazer isso, utiliza-se a função descrita abaixo:

```
digitalWrite(pino, modo);
```

Onde na opção pino, informa-se qual pino está fazendo acesso e na opção modo o estado que assumirá, neste caso em

nível alto, ou seja 5V ou nível baixo, neste caso 0V. Para nível alto deve-se escrever HIGH e para nível baixo LOW. Abaixo o código fonte completo.

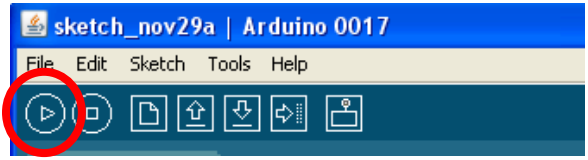
```
void setup()
{
     pinMode(13,OUTPUT);
}

void loop()
{
     digitalWrite(13, HIGH);
}
```

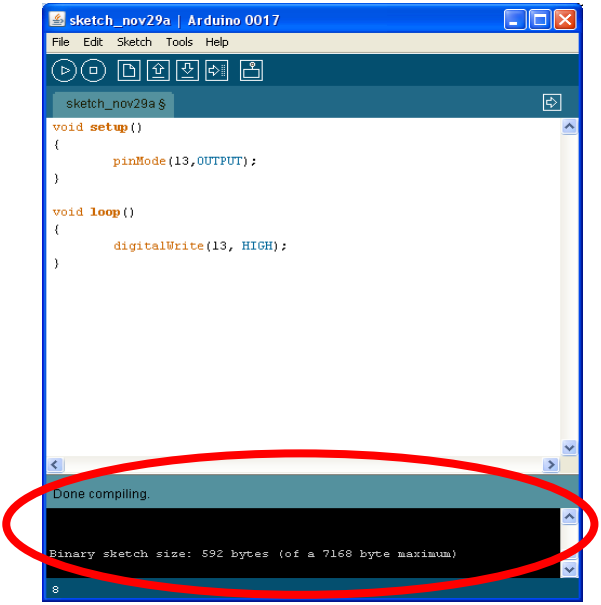


Neste instante para verificar se o código escrito está utiliza-se a ferramenta de compilação do Arduino. Vá ao menu Sketch ->

Verify/Compile ou pressione o botão marcado abaixo para iniciar a compilação.



Caso tudo esteja ok, a mensagem marcada abaixo irá aparecer.



Caso tal mensagem não apareça, revise o seu código e observe se está conforme apresentado.

Neste momento, salve o exemplo indo ao menu Arquivo -> Salvar e escolha a pasta que melhor lhe convier. Em seguida, realize o upload para a placa de modo a carregar o programa na placa Arduino.

Capítulo 4

Transmissão Serial

1. Introdução

A comunicação serial ainda é uma das mais usadas e neste capítulo é apresentado os passos para transmitir dados usando este tipo de interface. Será transmitida uma string, onde a placa Arduino UNO fornece os recursos para que possa enviar tal informação.

2. Programando o Arduino

Deve-se configurar a velocidade no qual é feita a comunicação, podendo esta ser de várias velocidades, como 1200 bps (bits por segundo), 2400 bps, 4800 bps, 9600 bps etc. Neste exemplo a velocidade usada é de 9600 bps. Abaixo está apresentado a função que configura a velocidade de comunicação:

```
Serial.begin(velocidade);
```

A função para enviar a string está apresentada a seguir:

```
Serial.println(texto);
```

O exemplo completo está apresentado logo abaixo enviando a string para o PC via RS232 a 9600 bps.

```
void setup()
{
  Serial.begin(9600);
  Serial.println("Aprendendo Arduino");
}

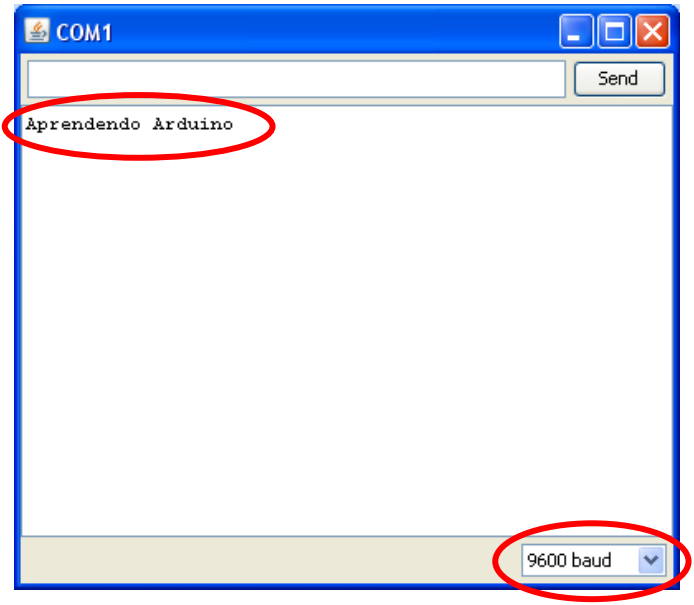
void loop()
{

}
```

Para visualizar os dados será utilizado o Serial Monitor, que já vem com o próprio software Arduino. Clique na opção marcada abaixo para inicializar este software.



Neste momento, a seguinte tela surgirá em que após a gravação da placa será possível visualizar o resultado apresentado abaixo a 9600 bps.

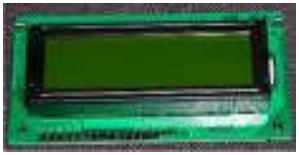

COM1
Send
Aprendendo Arduino
9600 baud

Capítulo 5

Display LCD

1. Introdução

Os displays do tipo LCD são muito usados para mostrar informações do estado de uma máquina. Na figura a seguir, é apresentado aspecto típico desta interface.

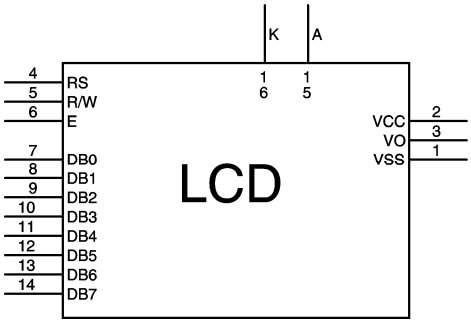


Existem vários tipos de display, como os de 2x20, que significa 2 linhas por 20 colunas, 4x20, 4x16 etc. Neste exemplo, usaremos um display do tipo 2x16, onde há duas linhas e 16 colunas.

O objetivo é apresentar um texto no LCD, de forma a entender como montar um programa no Arduino para comandá-lo.

2. Montando o Hardware

Na figura a seguir, é apresentado a pinagem típica de um display LCD:



Siga a tabela apresentada a seguir para ligar o display LCD a placa Arduino, de forma a apresentarmos o exemplo.

Display LCD	Placa Arduino
VCC	5V
VSS	GND
VO	Em série com resistor de 1kΩ ao GND
RS	Pino 8
EN	Pino 9
D4	Pino 10
D5	Pino 11
D6	Pino 12
D7	Pino 13

Os outros pinos não citados, não são usados.

3. Programando o Arduino

Novamente foi utilizado uma biblioteca para que o exemplo fique funcional, da mesma forma quando foi usado a memória EEPROM. A biblioteca é a LiquidCrystal.h e será apresentada a seguir no código fonte. Além disso, precisaremos usar a função lcd.begin para configurar o tipo de display conectado a placa Arduino. A seguir é apresentado tal função:

```
lcd.begin(colunas,linhas);
```

Em colunas, define-se o número de colunas no qual o display possui e em linhas o número de linhas. Feito isso, usa-se a função lcd.setCursor para posicionar onde iremos imprimir determinada informação. A seguir, tal função é apresentada:

```
lcd.setCursor(coluna,linha);
```

Onde caso coluna esteja em 0, a coluna 1 estará selecionada e caso esteja em 1, a coluna 2 e assim sucessivamente. Mesma ideia é válida para a opção linha. Finalmente, para imprimirmos um valor no LCD, usaremos a função lcd.print como apresentado a seguir:

```
lcd.print(texto);
```

Observe a seguir os passos para mostrar uma mensagem na 1º e 2º linha do display lcd.

```
#include <LiquidCrystal.h>
LiquidCrystal lcd(8, 9, 10, 11, 12, 13);

void setup()
{
  lcd.begin(16, 2);
  lcd.setCursor(0,0);
  lcd.print("Placa Arduino");
  lcd.setCursor(0,1);
  lcd.print("Exemplo LCD");
}

void loop()
{

}
```

Capítulo 6

Medindo Etanol em ppm

1. Sensor TGS 822

O sensor TGS 822, desenvolvido pela Figaro Sensor (www.figarosensor.com), permite medir uma série de vapores orgânicos como Metano, Monóxido de Carbono, Isobutano, Hexano, Benzeno, Etanol e Acetona. Cada um destas soluções, apresenta uma curva diferente de concentração em ppm (Parte Por Milhão), na faixa de 50 a 5000 ppm.

Este sensor apresenta alta sensibilidade, estabilidade, baixo custo e sua interface elétrica é relativamente simples. Suas principais aplicações são na detecção de etanol (podendo ser usado como bafômetro), detecção de vazamento de gás, detectores de solvente em fábrica etc. O elemento sensor é composto de Dióxido de Estanho (SnO_2), que varia sua condutividade de acordo com a concentração do gás presente em seu ambiente de instalação. Com o aumento da concentração de gás no ar, sua condutividade aumenta, em que desta forma um Arduino pode mensurar a concentração de determinado gás neste ambiente.

A próxima figura mostra a resposta de sensibilidade do sensor, de acordo com a concentração em ppm do solvente orgânico que estiver sendo medido pelo mesmo.

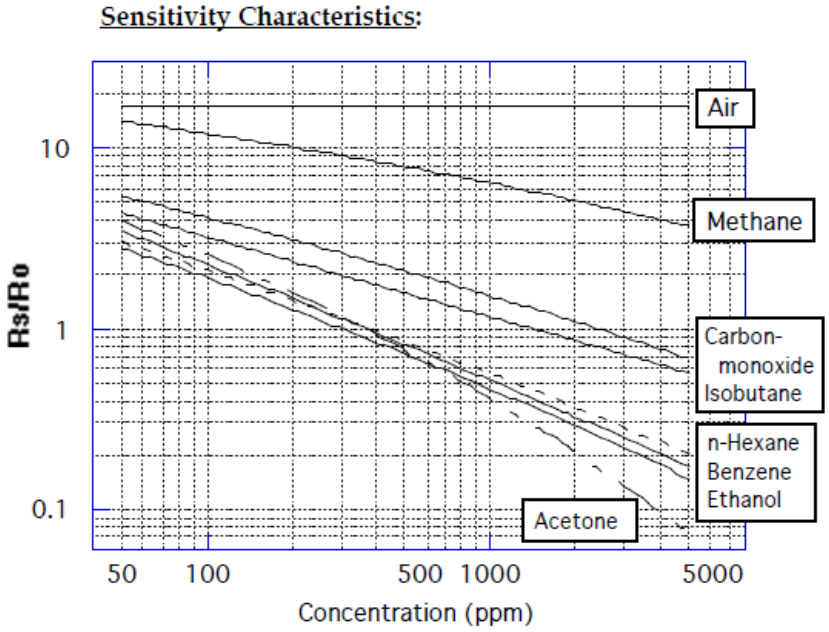


Figura 2.1: Resposta de acordo com a concentração de solvente
Fonte: TGS 822 Datasheet

A resposta do sensor varia de acordo com a temperatura e umidade relativa (R.H) do ambiente em que o mesmo estiver instalado, de acordo com a figura a seguir fornecida pelo próprio datasheet do componente.

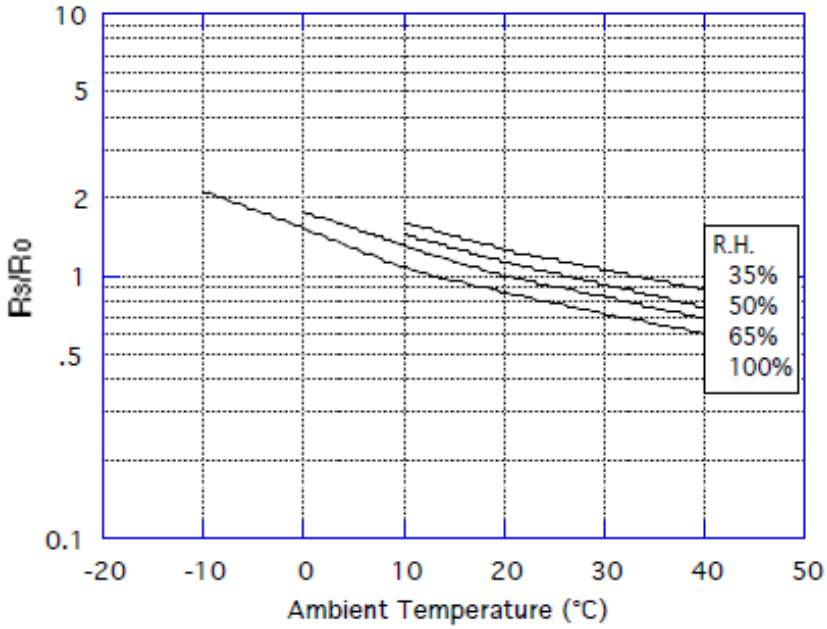


Figura 2.2: Influência da variação de umidade e temperatura
Fonte: TGS 822 Datasheet

Em determinados projetos que necessitem de maior precisão na resposta da medição de gases, pode ser necessário haver um sensor de umidade e temperatura, de modo a ajustar a resposta do TGS 822 de acordo com estas variações.

Este sensor é composto de 6 pinos, conforme pode ser visto na figura a seguir.

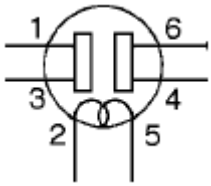


Figura 2.3: Pinagem do TGS 822
Fonte: TGS 822 Datasheet

A função da cada um dos pinos está descrita a seguir.

Pino	Função	Descrição
1 e 3	V_C	Alimentação positiva. Máximo de 24 Vdc.
2	V_H	Tensão de aquecimento. Tensão de 5V DC ou AC
5	GND	Terra da fonte
4 e 6	V_{RL}	Tensão variável de acordo com a concentração de gases

Fonte: TGS 822 Datasheet

Normalmente, os pinos 1, 2 e 3 ficam ligados juntos na tensão de 5 V. A saída V_{RL} fica acoplada a um resistor de carga (R_L), pois a resistência interna do sensor, ou seja, sua condutividade, varia segundo esta concentração, permitindo ao Arduino mensurá-la.

A próxima figura demonstra um circuito típico a ser montado para uso deste sensor.

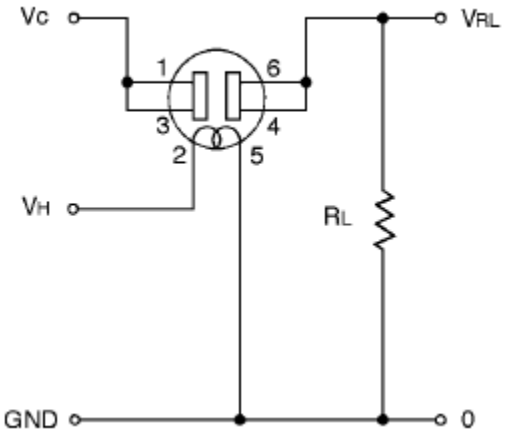


Figura 2.4: Circuito típico com TGS 822
Fonte: TGS 822 Datasheet

Um valor usual para R_L é de 10 kΩ. Observe pela curva do gráfico 2.1 que a concentração em ppm é encontrada de acordo com a relação R_S/R_0. O valor de R_S é definido como a resistência do sensor de acordo com a concentração de gás ambiente enquanto que R_0 para uma concentração de 50 ppm de Etanol, tipicamente 5.000Ω. O valor de R_S será encontrado segundo a fórmula a seguir.

$$R_S = \left(\frac{V_C}{V_{RL}} - 1 \right).R_L$$

Considerando V_C=5V e R_L=10.000Ω, o cálculo fica da seguinte forma.

$$R_S = \left(\frac{5}{V_{RL}} - 1 \right).10000$$

O valor de V_{RL} é encontrado através da leitura do conversor analógico do Arduino (CAD), estando na faixa de 0 a 5 V. Após encontrar R_S, basta encontrar a relação deste valor com R_0 (que vale 5.000Ω) para em seguida encontrar a concentração em ppm.

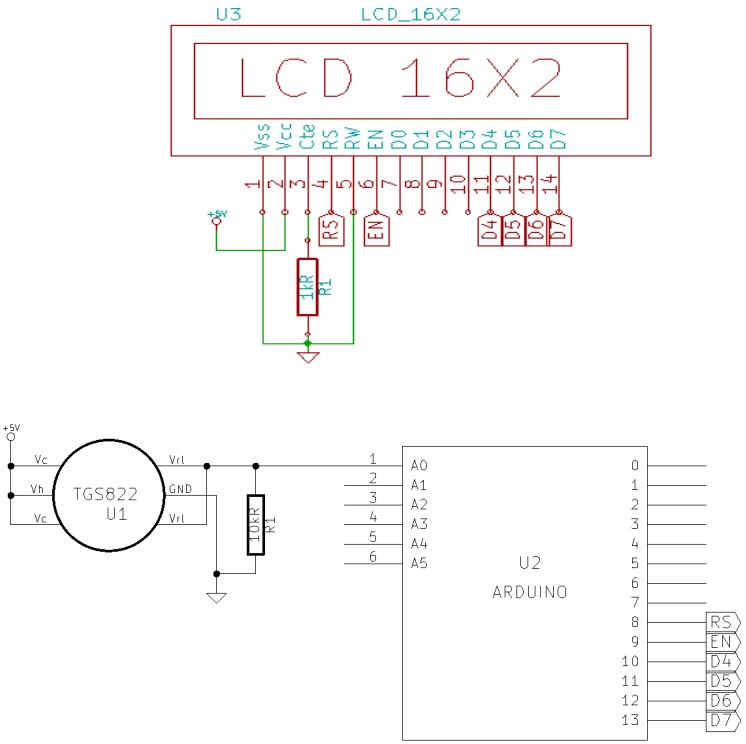
Chamando de *x* a relação entre R_S/R_0 e *ppm* a concentração, a função a seguir descreve como é encontrada a concentração.

$$ppm = 169.(\log(x))^4 - 147.(\log(x))^3 + 207.(\log(x))^2 - 496.\log(x) + 332$$

A descrição de como esta função foi encontrada está disponível no artigo de Jiana Shin, onde o link está disponível na seção Referências desta literatura.

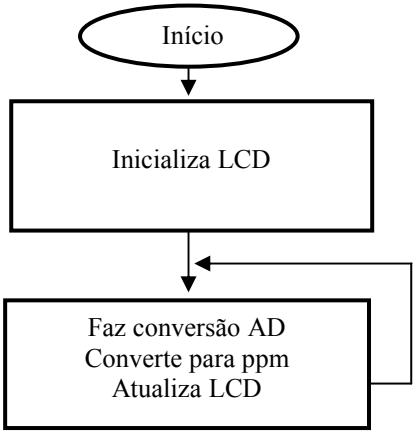
2. Esquema elétrico

O esquema elétrico para este exemplo está apresentado a seguir, onde observa-se o display LCD que mostra o valor da concentração em ppm de Etanol e o sensor que é a fonte de dados para o Arduino.



3. Fluxograma

O fluxograma que rege o funcionamento do programa está apresentado a seguir. Observe que inicialmente é feita a inicialização do LCD e no loop principal a leitura contínua da entrada analógica e em seguida a conversão para ppm, para que finalmente o LCD seja atualizado.



4. Código fonte

A seguir o código fonte em Arduino que mede a concentração em ppm do Etanol em determinado meio através do sensor TGS 822 e apresenta no LCD o resultado da medição em ppm. Após a leitura analógica do canal 0 encontra-se a relação R_S/R_0 para se aplicar a função apresentada, passando como parâmetro esta relação. O retorno da função é a concentração em ppm para que em seguida, este resultado seja apresentado no LCD.

```
/*  *  *  *  *  *  *  *  *  *  *  *  *
 *          Cerne Tecnologia          *
 *        www.cerne-tec.com.br        *
 *    Projetos em Arduino - Etanol    *
 *  *  *  *  *  *  *  *  *  *  *  *  */
#include <LiquidCrystal.h>
```

```
LiquidCrystal lcd(8,9,10,11,12,13);

/*  *  *  *  *  *  *  *  *  *  *  *  *
 *      Função de inicialização      *
 *  *  *  *  *  *  *  *  *  *  *  *  */
void setup(void)
{
  lcd.begin(16,2);
}

/*  *  *  *  *  *  *  *  *  *  *  *  *
 *          Função de loop           *
 *  *  *  *  *  *  *  *  *  *  *  *  */
void loop(void)
{
  lcd.clear();
  lcd.setCursor(4,0);
  lcd.print(le_ppm_alcool());
  lcd.print(" ppm");
  delay(1000);
}

/*  *  *  *  *  *  *  *  *  *  *  *  *
 *  Função para ler ppm do ambiente  *
 *  *  *  *  *  *  *  *  *  *  *  *  */
float le_ppm_alcool(void)
{
  float ppm,ad,vrl,rs,rs_ro,x;

  ad = analogRead(0);     //Lê AD de 0 a 1023
  vrl=(ad*5.0)/1023;      //Encontra Vrl
  rs=((5.0/vrl)-1)*10000; //Encontra Rs
  rs_ro=rs/5000;          //Encontra relação Rs/Ro
  x=log(rs_ro);           //Acha o log
  ppm=169*(x*x*x*x)-147*(x*x*x)+207*(x*x)-496*x+332;
                          //Equação para achar ppm
  return(ppm);            //Concentração em ppm
}
```